

RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000000000	FFFFFFFFFFF	FFFFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000000000	FFFFFFFFFFF	FFFFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000000000	FFFFFFFFFFF	FFFFFFFFFFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNNNNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNNNNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNNNNN	NNN	000	FFF	FFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000	FFFFFFFFF	FFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000	FFFFFFFFF	FFFFFFFFF
RRRRRRRRRRRR		UUU	UUU	NNN	NNN	000	FFFFFFFFF	FFFFFFFFF
RRR	RRR	UUU	UUU	NNN	NNNNNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNNNNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNNNNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUU	UUU	NNN	NNN	000	FFF	FFF
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	000000000	FFF	FFF	
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	000000000	FFF	FFF	
RRR	RRR	UUUUUUUUUUUUUUUU	NNN	NNN	000000000	FFF	FFF	

DDDDDDDD		SSSSSSSS	PPPPPPPP	EEEEEEEEEE	NN	NN	TTTTTTTTTT
DDDDDDDD		SSSSSSSS	PPPPPPPP	EEEEEEEEEE	NN	NN	TTTTTTTTTT
DD	DD	SS	PP	PP	NN	NN	TT
DD	DD	SS	PP	PP	NN	NN	TT
DD	DD	SS	PP	PP	NNNN	NN	TT
DD	DD	SS	PP	PP	NNNN	NN	TT
DD	DD		PPPPPPPP	EEEEEEEEEE	NN	NN	TT
DD	DD	SSSSSS	PPPPPPPP	EEEEEEEEEE	NN	NN	TT
DD	DD		PP	EE	NN	NNNN	TT
DD	DD		PP	EE	NN	NNNN	TT
DD	DD		PP	EE	NN	NN	TT
DD	DD		PP	EE	NN	NN	TT
DD	DD		PP	EE	NN	NN	TT
DDDDDDDD		SSSSSSSS	PP	EEEEEEEEEE	NN	NN	TT
DDDDDDDD		SSSSSSSS	PP	EEEEEEEEEE	NN	NN	TT

```

LL               IIIIII               SSSSSSSS
LL               IIIIII               SSSSSSSS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LL               II                   SSSSSS
LL               II                   SSSSSS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LL               II                   SS
LLLLLLLLLLLLLL  IIIIII               SSSSSSSS
LLLLLLLLLLLLLL  IIIIII               SSSSSSSS

```



```
0001 0 %TITLE 'Processes the several .DISPLAY <item> directives'
0002 0 MODULE dspent ( IDENT = 'V04-000'
P 0003 0      %BLISS32C, ADDRESSING_MODE (EXTERNAL  = LONG_RELATIVE,
0004 0      ) =
0005 0      NONEXTERNAL = LONG_RELATIVE)
0006 1 BEGIN
0007 1
0008 1 *****
0009 1 *
0010 1 *  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0011 1 *  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0012 1 *  ALL RIGHTS RESERVED.
0013 1 *
0014 1 *  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0015 1 *  ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0016 1 *  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0017 1 *  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0018 1 *  OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0019 1 *  TRANSFERRED.
0020 1 *
0021 1 *  THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0022 1 *  AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0023 1 *  CORPORATION.
0024 1 *
0025 1 *  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0026 1 *  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0027 1 *
0028 1 *
0029 1 *****
0030 1
0031 1
0032 1 ++
0033 1 FACILITY:      DSR (Digital Standard RUNOFF) / DSRPLUS
0034 1
0035 1 ABSTRACT:
0036 1
0037 1      Processes .DISPLAY EXAMPLE, .DISPLAY FIGURE, .DISPLAY TABLE,
0038 1      .DISPLAY APPENDIX, and .DISPLAY CHAPTER directives.
0039 1
0040 1 ENVIRONMENT:  Transportable
0041 1
0042 1 AUTHOR:      Keith Dawson      CREATION DATE: April 1982
0043 1
```

```

: 45      0044 1 %SBTTL 'Revision History'
: 46      0045 1   MODIFIED BY:
: 47      0046 1
: 48      0047 1       008      KFA00008      Ken Alden      01-Aug-1983
: 49      0048 1       Fixed bug: .DISPLAY APPENDIX never got added during work
: 50      0049 1         of #007.
: 51      0050 1
: 52      0051 1       007      KAD00007      Keith Dawson   09-May-1983
: 53      0052 1       Fixed bug: .DISPLAY {anything} was overwriting the NPAGEN
: 54      0053 1         value of the (chapter) display code. Missing BEGIN-END.
: 55      0054 1
: 56      0055 1       006      REM00006      Ray Marshall   27-April-1983
: 57      0056 1       Conditionalized this module so that it can be compiled for DSR
: 58      0057 1         without trying to bring in unneeded logic or use symbols
: 59      0058 1         unavailable to DSR. DSR now uses this module to process the
: 60      0059 1         .DISPLAY APPENDIX and .DISPLAY CHAPTER directives.
: 61      0060 1
: 62      0061 1       005      RER00005      Ron Randall    19-Apr-1983
: 63      0062 1       Fixed bug in capturing pre/poststrings on entities.
: 64      0063 1
: 65      0064 1       004      KFA00004      Ken Alden      30-Mar-1983
: 66      0065 1       Fixed bug in displaying chapter page numbers
: 67      0066 1
: 68      0067 1       003      KFA00003      Ken Alden      07-Mar-1983
: 69      0068 1       Global edit of all modules. Updated module names, idents,
: 70      0069 1       copyright dates. Changed require files to BLISS library.
: 71      0070 1
: 72      0071 1

```


Processes the several .DISPLAY <item> directive
Module Level Declarations

VAX-11 BLISS-32 V4.0-742
[RUNOFF.SRC]DSPENT.BLI:1

Page 3
(3)

:	Ro
:	1
:	1

```

74      0072 1 %SBTTL 'Module Level Declarations'
75      0073 1
76      0074 1 | TABLE OF CONTENTS:
77      0075 1 |
78      0076 1 FORWARD ROUTINE
79      0077 1     DSPENT : NOVALUE;
80      0078 1 |
81      0079 1 | INCLUDE FILES:
82      0080 1 |
83      0081 1 LIBRARY 'NXPORT:XPORT';           ! XPORT Library
84      0082 1 REQUIRE 'REQ:RNODEF';           ! RUNOFF variant definitions
85      0213 1
86      U 0214 1 %IF DSRPLUS %THEN
87      U 0215 1 LIBRARY 'REQ:DPLLIB';           ! DSRPLUS BLISS Library
88      0216 1 %ELSE
89      0217 1 LIBRARY 'REQ:DSRLIB';           ! DSR BLISS Library
90      0218 1 %FI
91      0219 1
92      0220 1 |
93      0221 1 | OWN STORAGE:
94      0222 1 |
95      0223 1 OWN
96      0224 1     fs_allocate (e_pre_string, 10),
97      0225 1     fs_allocate (e_post_string, 10),
98      0226 1     fs_allocate (f_pre_string, 10),
99      0227 1     fs_allocate (f_post_string, 10),
100     0228 1     fs_allocate (t_pre_string, 10),
101     0229 1     fs_allocate (t_post_string, 10);
102     0230 1 |
103     0231 1 | EXTERNAL REFERENCES:
104     0232 1 |
105     0233 1 EXTERNAL
106     0234 1     ecc           : $ecc_blockvector,
107     0235 1     pagen        : page_definition,
108     0236 1     npagen       : page_definition,
109     0237 1     hllist       : counted_list,
110     0238 1     ira         : fixed_string;
111     0239 1
112     0240 1 EXTERNAL LITERAL
113     0241 1     rnfqst;
114     0242 1
115     0243 1 EXTERNAL ROUTINE
116     0244 1     erms,
117     0245 1     getdd,
118     0246 1     getqs,
119     0247 1     rskips,
120     0248 1     skpsep;
121     0249 1

```

```
123 0250 1 %SBTTL 'DSPENT -- main routine'
124 0251 1 GLOBAL ROUTINE dspent (HANDLER) : NOVALUE =
125 0252 1
126 0253 1 ++
127 0254 1 FUNCTIONAL DESCRIPTION:
128 0255 1
129 0256 1 Processes .DISPLAY EXAMPLE, FIGURE, TABLE, CHAPTER, and APPENDIX
130 0257 1 directives.
131 0258 1
132 0259 1 FORMAL PARAMETERS:
133 0260 1
134 0261 1 HANDLER - Indicates which command is to be processed.
135 0262 1
136 0263 1 IMPLICIT INPUTS: None
137 0264 1
138 0265 1 IMPLICIT OUTPUTS:
139 0266 1
140 0267 1 Updates the appropriate field in the ECC blockvector structure.
141 0268 1
142 0269 1 ROUTINE VALUE:
143 0270 1 COMPLETION CODES: None
144 0271 1
145 0272 1 SIDE EFFECTS: None
146 0273 1 --
147 0274 1
148 0275 2 BEGIN
149 0276 2 LOCAL
150 0277 2 getdd_result,
151 0278 2 get_pre_result,
152 0279 2 get_post_result,
153 0280 2 display_code,
154 0281 2 offset;
155 0282 2
156 0283 3 offset = (SELECTONE .handler OF
157 0284 3 SET
158 U 0285 3 %IF dsrplus %THEN
159 UU 0286 3 [h_display_examp] : examp_offset;
160 UU 0287 3 [h_display_figur] : figur_offset;
161 U 0288 3 [h_display_table] : table_offset;
162 0289 3 %FI
163 0290 3 [h_display_chapt] : chap_offset;
164 0291 3 [h_display_appen] : append_offset;
165 U 0292 3 %IF dsrplus %THEN
166 U 0293 3 [h_display_head] : hcoll_offset + .hllist [cl_index];
167 0294 3 %FI
168 0295 2 TES);
169 0296 2
170 0297 2 fs_init (e_pre_string);
171 0298 2 fs_init (e_post_string);
172 0299 2 fs_init (f_pre_string);
173 0300 2 fs_init (f_post_string);
174 0301 2 fs_init (t_pre_string);
175 0302 2 fs_init (t_post_string);
176 0303 2
177 U 0304 2 %IF dsrplus %THEN
178 U 0305 2 ! Skip spaces and tabs before the optional quoted string.
179 U 0306 2 rskips (ira);
```



```
180      U 0307 2
181      U 0308 2      ! Try to get a quoted string.
182      U 0309 2      get_pre_result = (SELECTONE .handler OF
183      U 0310 2          SET
184      U 0311 2          [h_display_examp] : getqs (e_pre_string);
185      U 0312 2          [h_display_figur] : getqs (f_pre_string);
186      U 0313 2          [h_display_table] : getqs (t_pre_string);
187      U 0314 2          TES);
188      U 0315 2
189      U 0316 2      ! Report error if specified string exceeded 5 characters.
190      U 0317 2      IF .get_pre_result EQL getqs_too_long
191      U 0318 2      THEN
192      U 0319 2          BEGIN
193      U 0320 2              SELECTONE .HANDLER OF
194      U 0321 2                  SET
195      U 0322 2                  [h_display_examp] :
196      U 0323 2                      erms (rnfqst, .fs_start (e_pre_string), .fs_length (e_pre_string));
197      U 0324 2                  [h_display_figur] :
198      U 0325 2                      erms (rnfqst, .fs_start (f_pre_string), .fs_length (f_pre_string));
199      U 0326 2                  [h_display_table] :
200      U 0327 2                      erms (rnfqst, .fs_start (t_pre_string), .fs_length (t_pre_string));
201      U 0328 2                  TES;
202      U 0329 2              END;
203      U 0330 2
204      U 0331 2
205      U 0332 2
206      U 0333 2
207      U 0334 2
208      U 0335 2      %FI
209      U 0336 2      ! Skip a comma, spaces and tabs before the display descriptor.
210      U 0337 2      skipsep (ira);
211      U 0338 2
212      U 0339 2      ! Get the display descriptor.
213      U 0340 2      getdd_result = getdd (display_code);
214      U 0341 2
215      U 0342 2      ! Distinguish between missing display code and one that is given.
216      U 0343 2      IF .getdd_result EQL 0
217      U 0344 2      ! No display code supplied. Use the standard display as default.
218      U 0345 2      THEN
219      U 0346 2          display_code = tconvrt_dec_noz;
220      U 0347 2
221      U 0348 2      %IF dsrplus %THEN
222      U 0349 2      ! Skip a comma, spaces and tabs before the second optional quoted string.
223      U 0350 2      skipsep (ira);
224      U 0351 2
225      U 0352 2      ! Try to get a quoted string.
226      U 0353 2      get_post_result = (SELECTONE .handler OF
227      U 0354 2          SET
228      U 0355 2          [h_display_examp] : getqs (e_post_string);
229      U 0356 2          [h_display_figur] : getqs (f_post_string);
230      U 0357 2          [h_display_table] : getqs (t_post_string);
231      U 0358 2          TES);
232      U 0359 2
233      U 0360 2      ! Report error if specified string exceeded 5 characters.
234      U 0361 2      IF .get_post_result EQL getqs_too_long
235      U 0362 2      THEN
236      U 0363 2          BEGIN
```

```

237 U 0364 2      SELECTONE .handler OF
238 U 0365 2      SET
239 U 0366 2      [h_display_examp] :
240 U 0367 2      erms (rnfqst, .fs_start (e_post_string), .fs_length (e_post_string));
241 U 0368 2
242 U 0369 2      [h_display_figur] :
243 U 0370 2      erms (rnfqst, .fs_start (f_post_string), .fs_length (f_post_string));
244 U 0371 2
245 U 0372 2      [h_display_table] :
246 U 0373 2      erms (rnfqst, .fs_start (t_post_string), .fs_length (t_post_string));
247 U 0374 2
248 U 0375 2      TES;
249 U 0376 2      END;
250 U 0377 2      !+
251 U 0378 2      Store the string(s) and descriptor gotten, if any.
252 U 0379 2      !-
253 U 0380 2      IF (.get_pre_result EQL getqs_normal) OR
254 U 0381 2      (.get_pre_result EQL getqs_too_long)
255 U 0382 2      THEN
256 U 0383 2      BEGIN
257 U 0384 2      SELECTONE .handler OF
258 U 0385 2      SET
259 U 0386 2
260 U 0387 2      [h_display_examp] :
261 U 0388 2      BEGIN
262 U 0389 2      ecc [.offset, ecc$a_pre_ptr] = .fs_start (e_pre_string);
263 U 0390 2      ecc [.offset, ecc$h_pre_len] = .fs_length (e_pre_string);
264 U 0391 2      END;
265 U 0392 2
266 U 0393 2      [h_display_figur] :
267 U 0394 2      BEGIN
268 U 0395 2      ecc [.offset, ecc$a_pre_ptr] = .fs_start (f_pre_string);
269 U 0396 2      ecc [.offset, ecc$h_pre_len] = .fs_length (f_pre_string);
270 U 0397 2      END;
271 U 0398 2
272 U 0399 2      [h_display_table] :
273 U 0400 2      BEGIN
274 U 0401 2      ecc [.offset, ecc$a_pre_ptr] = .fs_start (t_pre_string);
275 U 0402 2      ecc [.offset, ecc$h_pre_len] = .fs_length (t_pre_string);
276 U 0403 2      END;
277 U 0404 2      TES;
278 U 0405 2      END;
279 U 0406 2      XFI
280 U 0407 2      IF .getdd_result EQL 1
281 U 0408 2      THEN
282 U 0409 2      BEGIN
283 U 0410 2      ecc [.offset, ecc$h_display_desc] = .display_code;
284 U 0411 2
285 U 0412 2      IF .offset EQL chap_offset
286 U 0413 2      THEN
287 U 0414 2      BEGIN
288 U 0415 2      pagen [sct_chapt_d] = .display_code;
289 U 0416 2      npagen [sct_chapt_d] = .display_code;
290 U 0417 2      END;
291 U 0418 2
292 U 0419 2      IF .offset EQL append_offset
293 U 0420 2      THEN

```



```

: 294      0421  4      BEGIN
: 295      0422  4      pagen [sct_appen_d] = .display_code;
: 296      0423  4      npagen [sct_appen_d] = .display_code;
: 297      0424  3      END;
: 298      0425  2      END;
: 299      0426  2
: 300      U 0427  2      %IF dsrplus %THEN
: 301      UU 0428  2      IF (.get_post_result EQL getqs_normal) OR
: 302      UU 0429  2      (.get_post_result EQL getqs_too_long)
: 303      UU 0430  2      THEN
: 304      UU 0431  2      BEGIN
: 305      UU 0432  2      SELECTONE .handler OF
: 306      UU 0433  2      SET
: 307      UU 0434  2
: 308      UU 0435  2      [h_display_examp] :
: 309      UU 0436  2      BEGIN
: 310      UU 0437  2      ecc [.offset, ecc$a_post_ptr] = .fs_start (e_post_string);
: 311      UU 0438  2      ecc [.offset, ecc$h_post_len] = .fs_length (e_post_string);
: 312      UU 0439  2      END;
: 313      UU 0440  2
: 314      UU 0441  2      [h_display_figur] :
: 315      UU 0442  2      BEGIN
: 316      UU 0443  2      ecc [.offset, ecc$a_post_ptr] = .fs_start (f_post_string);
: 317      UU 0444  2      ecc [.offset, ecc$h_post_len] = .fs_length (f_post_string);
: 318      UU 0445  2      END;
: 319      UU 0446  2
: 320      UU 0447  2      [h_display_table] :
: 321      UU 0448  2      BEGIN
: 322      UU 0449  2      ecc [.offset, ecc$a_post_ptr] = .fs_start (t_post_string);
: 323      UU 0450  2      ecc [.offset, ecc$h_post_len] = .fs_length (t_post_string);
: 324      UU 0451  2      END;
: 325      UU 0452  2      TES;
: 326      U 0453  2      END;
: 327      0454  2      %FI
: 328      0455  1      END;

```

! End of DSPENT

```

.TITLE DSPENT Processes the several .DISPLAY <item> di
       rective
.IDENT \V04-000\
.PSECT $OWNS$,NOEXE,2

```

00000000	0000000A	00000000	00000000	00000	E_PRE_STRING:		
				00010	.LONG	0, 0, 10, 0	:
				0001A	.BLKB	10	
				0001C	.BLKB	2	
00000000	0000000A	00000000	00000000	0001C	E_POST_STRING:		
				0002C	.LONG	0, 0, 10, 0	:
				00036	.BLKB	10	
				00038	.BLKB	2	
00000000	0000000A	00000000	00000000	00038	F_PRE_STRING:		
				00048	.LONG	0, 0, 10, 0	:
				00052	.BLKB	10	
				00054	.BLKB	2	
00000000	0000000A	00000000	00000000	00054	F_POST_STRING:		
					.LONG	0, 0, 10, 0	:

```
00000000 0000000A 00000000 00000000 00064 .BLKB 10
0006E .BLKB 2
00070 T_PRE_STRING: .LONG 0, 0, 10, 0
00080 .BLKB 10
0008A .BLKB 2
0008C T_POST_STRING: .LONG 0, 0, 10, 0
0009C .BLKB 10

.EXTRN ECC, PAGEN, NPAGEN
.EXTRN HLLIST, IRA, RNFOST
.EXTRN ERMS, GETDD, GETQS
.EXTRN RSKIPS, SKPSEP

.PSECT $CODE$,NOWRT,2

.ENTRY DSPENT, Save R2,R3
MOVAB E_PRE_STRING, R3
SUBL2 #2, SP
MOVL HANDLER, R0
CML R0, #27
BNEQ 1$
MOVL #10, OFFSET
BRB 3$
1A 50 D1 0001A 1$: CML R0, #26
52 05 13 0001D BEQL 2$
52 01 CE 0001F MNEGL #1, OFFSET
52 03 11 00022 BRB 3$
52 0B D0 00024 2$: MOVL #11, OFFSET
63 0C A3 D4 00027 3$: CLRL E_PRE_STRING+12
04 63 10 A3 9E 0002A MOVAB E_PRE_STRING+16, E_PRE_STRING
1C A3 28 A3 D0 0002E MOVL E_PRE_STRING, E_PRE_STRING+4
20 A3 1C A3 D4 00032 CLRL E_POST_STRING+12
38 A3 44 A3 9E 00035 MOVAB E_POST_STRING+16, E_POST_STRING
3C A3 38 A3 D0 00047 MOVL E_POST_STRING, E_POST_STRING+4
54 A3 60 A3 D4 0004C CLRL F_PRE_STRING+12
58 A3 64 A3 9E 0004F MOVAB F_PRE_STRING+16, F_PRE_STRING
70 A3 7C A3 D0 00054 MOVL F_PRE_STRING, F_PRE_STRING+4
74 A3 70 A3 D4 00059 CLRL F_POST_STRING+12
008C C3 0080 C3 9E 0005C MOVAB F_POST_STRING+16, F_POST_STRING
0090 C3 0098 C3 D0 00062 MOVL F_POST_STRING, F_POST_STRING+4
00000000G EF 00000000G C3 D4 00067 CLRL T_PRE_STRING+12
00000000G EF 00000000G C3 9E 0006B MOVAB T_PRE_STRING+16, T_PRE_STRING
00000000G EF 00000000G C3 D0 00072 MOVL T_PRE_STRING, T_PRE_STRING+4
00000000G EF 00000000G EF 9F 00079 PUSHAB IRA
00000000G EF 00000000G 01 FB 0007F CALLS #1, SKPSEP
00000000G EF 00000000G 5E DD 00086 PUSHL SP
00000000G EF 00000000G 01 FB 00088 CALLS #1, GETDD
00000000G EF 00000000G 50 D5 0008F TSTL GETDD_RESULT
00000000G EF 00000000G 02 12 00091 BNEQ 4$
00000000G EF 00000000G 6E D4 00093 CLRL DISPLAY CODE
00000000G EF 00000000G 50 D1 00095 4$: CML GETDD_RESULT, #1
00000000G EF 00000000G 3F 12 00098 BNEQ 6$
```


DSPP
V04-

[illegible]

```

: 329      0456 1
: 330      0457 1 END
: 331      0458 0 ELUDOM

```

! End of module

.....

.....

.....

.....

:

:

```

; Size:          218 code + 166 data bytes
; Run Time:      00:06.9
; Elapsed Time:  00:20.1
; Lines/CPU Min: 3971

```

Processes the several
DSPENT -- main routine

Processes the several .DISPLAY <item> directive N 12
DSPENI == main routine 16-Sep-1984 00:21:20

Page 10

```
: Lexemes/CPU-Min: 23531
: Memory Used: 71 pages
: Compilation Complete
```

DSPPA
V04-(

[illegible]

0339

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY